

07-10-00

A

WILLIAMS, MORGAN & AMERSON, P.C.

7676 HILLMONT, SUITE 250, HOUSTON, TX 77040
(713) 934-7000 FAX (713) 934-7011

Danny L. Williams
Terry D. Morgan
J. Mike Amerson
Kenneth D. Goodman
Barbara S. Kitchell, Ph.D.
Jeffrey A. Pyle
Ruben S. Bains
Jaion C. John
Randall C. Furlong, Ph.D.

Scott F. Diring*
George J. Oehling*
Shelley P.M. Fussey, Ph.D.*
Mark D. Moore, Ph.D.*
Carolanne M. King*
Louis H. Iselin, Ph.D.*
Raymund F. Eich, Ph.D.*

*Patent Agent

EXPRESS MAIL MAILING LABEL

NUMBER EL 522 492 376

DATE OF DEPOSIT 7TH DAY OF JULY, 2000

I hereby certify that this paper or fee is being deposited with the United States Postal Service with sufficient postage "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to: Assistant Commissioner for Patents, Washington D.C. 20231.

Kathy Jones
Signature

July 7, 2000

FILE 2007.0012900

GEORGE J. OEHLING, PATENT AGENT
713-934-4058

BOX PATENT APPLICATION
Assistant Commissioner for Patents
Washington, DC 20231

RE: *U.S. Patent Application Entitled "METHOD AND APPARATUS FOR PROVIDING ENHANCED ACCESS TO A LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL (LDAP) DIRECTORY SERVER"*

Inventor(s): David J. Lindner
Client Reference: P00-2935

Sir:

Transmitted herewith for filing are:

- (1) 18-page patent specification with 20 claims and an abstract (also Figures 1-4 on 4 sheets);
- (2) Declaration;
- (3) Assignment and Assignment Cover Sheet;
- (4) Power of Attorney; and

07/07/00
Jc878 U.S. PTO

Jc878 U.S. PTO
09/611920
07/07/00

07/07/00
Jc878 U.S. PTO

WILLIAMS, MORGAN & AMERSON, P.C.

Assistant Commissioner for Patents

July 7, 2000

Page 2

(5) Our check in the amount of the total filing fee (listed below).

All correspondence, notices, official letters and other communications should be directed to George J. Oehling, Patent Agent, Williams, Morgan & Amerson, P.C., 7676 Hillmont, Suite 250, Houston, TX 77040, and all telephone calls should be directed to George J. Oehling, Patent Agent at (713) 934-4058.

FILING FEE CALCULATION

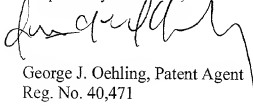
FOR		Small Entity	Large Entity
Total Claims	20 - 20 = 0	x \$9 = \$	or x \$18 = \$ 0.00
Independent Claims	4 - 3 = 1	x \$39 = \$	or x \$78 = \$ 78.00
Multiple Dependent Claim(s)		+ \$130 = \$	or + \$260 = \$ 0.00
Basic Fee:		+ \$345 = \$	or + \$690 = \$ 690.00
Assignment Recording Fee:	(\$40 per assignee)	+ = \$	+ = \$ 40.00
TOTAL FILING FEES		\$ <u>0.00</u>	\$ <u>808.00</u>

Pursuant to 37 C.F.R. § 1.10 the Applicant requests that the Patent and Trademark Office accept this application and accord a serial number and filing date as of the date this application is deposited with the U.S. Postal Service for Express Mail.

If the check is inadvertently omitted, or should any additional fees under 37 C.F.R. §§ 1.16 to 1.21 be required for any reason relating to the enclosed materials, or should an overpayment be included herein, the Assistant Commissioner is authorized to deduct or credit said fees from or to Williams, Morgan & Amerson, P.C. Deposit Account No. 50-0786/2007.012900.

Please date stamp and return the enclosed postcard to evidence receipt of these materials.

Respectfully submitted,



George J. Oehling, Patent Agent
Reg. No. 40,471

GJO/kd

Encl.

Application for United States Letters Patent

for

**METHOD AND APPARATUS FOR PROVIDING ENHANCED ACCESS
TO A LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL (LDAP)
DIRECTORY SERVER**

by

David J. Lindner

EXPRESS MAIL MAILING LABEL

NUMBER EL 522 492 376 US

DATE OF DEPOSIT JULY 7, 2000

I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to: Assistant Commissioner for Patents, Washington D.C. 20231.

Kathy Almas

Signature

METHOD AND APPARATUS FOR PROVIDING ENHANCED ACCESS TO A LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL (LDAP) DIRECTORY SERVER

BACKGROUND OF THE INVENTION

1. FIELD OF THE INVENTION

This invention relates generally to directory server access in computer systems, and, more particularly, to a method and apparatus for directory server access using a lightweight directory access protocol (LDAP) caching daemon.

2. DESCRIPTION OF THE RELATED ART

Lightweight Directory Access Protocol (LDAP) is an industry-standard software protocol that enables a person to locate organizations, individuals, as well as other resources such as files and devices, for example, within a network. The network may be the Internet, for example, or on a smaller scale, the network may be a corporate intranet. LDAP is essentially a "lightweight" version of Directory Access Protocol (DAP), which is part of X.500, a standard for directory services in a network. An advantage of LDAP is that it runs directly over Transmission Control Protocol/Internet Protocol (TCP/IP) and provides most of the functionality of DAP, however, at a much lower cost.

A directory provides information as to the location of a particular thing within the network. For example, on TCP/IP networks, which include the Internet, the Domain Name System (DNS) is the directory system used to relate the domain name to a specific network address (*i.e.*, a unique location in the network). LDAP allows a person to search for an individual, for example, without knowing the domain name or where that individual may be located within the network.

5 An LDAP directory is organized in a hierarchical tree-like structure that typically reflects political, geographical and/or organizational boundaries. For example, the directory may include countries at the top of the tree. These countries may then branch out into organizations, which may extend to organizational units, such as divisions, departments, *etc.*, and then onto individuals, such as people, files, documents and shared resources such as printers, for example.

10 An LDAP directory may be distributed among many servers, where each server may have a replicated version of the entire LDAP directory. An LDAP directory server, also commonly known as a Directory System Agent (DSA), typically receives a request for information from the LDAP directory from a client server that is running a particular application.

15 Turning now to the drawings, and specifically referring to Figure 1, a system 100 for providing LDAP directory server access to a plurality of client server applications is shown in accordance with the prior art. Typically, when a client server application 120 desires to access data from a directory server 110, the application 120 establishes a direct connection 20 125 to the directory server 110 through a binding operation. The function of the binding operation is to initiate a protocol session between a client and the directory server 110, and to allow authentication of the client to the server 110. The client server application 120 establishes the connection 125 to the server 110 via a bind request. Upon receiving the bind request from the client server application 120, the directory server 110 will authenticate the requesting client if necessary, and attempt to set up a protocol session with the client. The 25 directory server 110 subsequently sends a bind response to the client server application 120,

thereby providing an indication of the status of the session startup request. Upon
successfully establishing the connection 225 with the directory server 110, the application
120 then retrieves the desired data from the directory server 110 by performing a search
operation. After retrieving this desired data, the client server application 120 may then
5 perform an unbind operation to terminate the protocol session between the client server
application 120 and the directory server 110. Subsequent to receiving the unbind request, the
directory server 110 closes the connection 225 with the client server application 120.

10 A drawback currently encountered in LDAP, however, is that it requires each client to
bind directly to the directory server 110 before being able to perform a request for
information from the LDAP directory. That is, the client server applications 120 running on
the OS establish a direct connection to the directory server 110, thereby placing a substantial
load on the directory server 110 by engaging individual connections with each application
120. As a result of this additional load, the directory server 110 suffers substantial
15 degradation in performance for data retrieval.

The present invention is directed to overcoming, or at least reducing the effects of,
one or more of the problems set forth above.

SUMMARY OF THE INVENTION

In one aspect of the present invention, a method is provided for accessing a directory server. The method includes establishing a first plurality of connections between the directory server and a caching daemon. It is determined if an application is requesting information from the directory server. In response to determining that the application has requested information, it is determined if the requested information is stored in the caching daemon. The requested information is then sent to the application.

In another aspect of the present invention, an apparatus is provided. The apparatus comprises a directory server for storing information and a caching daemon. The caching daemon being adapted to establish a first plurality of connections to the directory server and determine if an application is requesting information from the directory server. The caching daemon is further adapted to determine if the requested information is stored within the caching daemon and to send the requested information to the application.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention may be understood by reference to the following description taken in conjunction with the accompanying drawings and appendices, in which like reference numerals identify like elements, and in which:

Figure 1 is a simplified diagram illustrating the access of a directory server by a plurality of client server applications according to the prior art;

Figure 2 illustrates a simplified diagram for providing access to the directory server from the plurality of client server applications via a Lightweight Directory Access Protocol (LDAP) caching daemon according to one embodiment of the present invention;

Figure 3 provides a more detailed representation of the LDAP caching daemon provided in Figure 2; and

Figure 4 illustrates a process for facilitating directory server access by the plurality of client server applications in accordance with one embodiment of the present invention.

While the invention is susceptible to various modifications and alternative forms, specific embodiments thereof have been shown by way of example in the drawings and are herein described in detail. It should be understood, however, that the description herein of specific embodiments is not intended to limit the invention to the particular forms disclosed, but on the contrary, the intention is to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the invention as defined by the appended claims.

DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

Illustrative embodiments of the invention are described below. In the interest of clarity, not all features of an actual implementation are described in this specification. It will of course be appreciated that in the development of any such actual embodiment, numerous implementation-specific decisions must be made to achieve the developers' specific goals, such as compliance with system-related and business-related constraints, which will vary from one implementation to another. Moreover, it will be appreciated that such a development effort might be complex and time-consuming, but would nevertheless be a routine undertaking for those of ordinary skill in the art having the benefit of this disclosure.

Referring back to the drawings, and specifically to that of Figure 2, a system 200 for facilitating access to a directory server 110 from a plurality of applications 120 running on an operating system (OS) is provided. In accordance with the illustrated embodiment, the applications 120 are generated by a plurality of client servers (not shown). According to one embodiment of the present invention, the directory server 110 operates in accordance with a Lightweight Directory Access Protocol (LDAP), which is an industry-standard network protocol for directory server access. It will be appreciated, however, that the directory server 110 may also operate according to various other industry-standard network protocols, such as Directory Access Protocol (DAP), for example, without departing from the spirit and scope of the present invention.

As previously discussed, the client server applications 120 running on the OS typically establish a direct connection to the directory server 110, thereby placing a substantial load on the directory server 110 by engaging individual connections with each

application 120. As a result of this additional load, the directory server 110 suffers substantial degradation in performance for data retrieval. To alleviate the burden on the directory server 110, an LDAP caching daemon 210 is provided for connection to the directory server 110 in accordance with one embodiment of the present invention. In the illustrated embodiment, the LDAP caching daemon 210 is a multi-threaded Internet UNIX daemon, and accesses data from the directory server 110 via a plurality of connections represented at 215. The number of connections at 215 may be configurable to facilitate the retrieval of data from the directory server 110 by the caching daemon 210. Furthermore, in addition to being coupled to the one directory server 110 provided in Figure 2, it will be appreciated that the caching daemon 210 may also couple to multiple directory servers 110 without departing from the spirit and scope of the present invention. Only one directory server 110, however, is depicted in Figure 2 for simplicity sake in illustrating the present invention. The LDAP directory server(s) 115 access data from an LDAP directory (not shown).

In accordance with the illustrated embodiment, the LDAP caching daemon 210 resides between the directory server 110 and a Security Integration Architecture (SIA) layer 220 of the OS. The plurality of client server applications 120 that run in the SIA layer 220 of the OS send data requests that are intended for the directory server 110 to the LDAP caching daemon 210. These applications 120 in the SIA layer 220 may include, file transfer protocol (ftp), telnet, rlogin, and CDE, for example. It will be appreciated, however, that the applications 120 may include various other application types running on the client server, and, thus, need not necessarily be limited to the aforementioned examples. Furthermore, the number of applications 120 that are able to access the caching daemon 210 may vary depending on system requirements.

10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
105
110
115
120
125
130
135
140
145
150
155
160
165
170
175
180
185
190
195
200
205
210
215
220
225
230
235
240
245
250
255
260
265
270
275
280
285
290
295
300
305
310
315
320
325
330
335
340
345
350
355
360
365
370
375
380
385
390
395
400
405
410
415
420
425
430
435
440
445
450
455
460
465
470
475
480
485
490
495
500
505
510
515
520
525
530
535
540
545
550
555
560
565
570
575
580
585
590
595
600
605
610
615
620
625
630
635
640
645
650
655
660
665
670
675
680
685
690
695
700
705
710
715
720
725
730
735
740
745
750
755
760
765
770
775
780
785
790
795
800
805
810
815
820
825
830
835
840
845
850
855
860
865
870
875
880
885
890
895
900
905
910
915
920
925
930
935
940
945
950
955
960
965
970
975
980
985
990
995

In one embodiment of the present invention, the connections 215 established between the directory server 110 and the caching daemon 210 run continuously awaiting for a connection 225 to be established between the client server applications 120 in the SIA layer 220 and the caching daemon 210. When a connection 225 is established between the application 120 and the caching daemon 210, a thread is created to handle the connection 225. Depending upon the operation being performed by the client server application 120, the thread may persist to perform multiple operations on behalf of the SIA layer 220, or may terminate upon completing a single operation by the application 120. It will be appreciated that the establishment of the thread may be performed by the “boss/worker” model, for example, where the “boss” listens for the connection 225 between the application 120 and the caching daemon 210 and then creates the thread once the connection 225 is established. Establishing threads in accordance with the “boss/worker” model is well known to those of ordinary skill in the art. Accordingly, the specifics for establishing the threads will not be discussed herein to avoid unnecessarily obscuring the present invention.

According to one embodiment of the present invention, the caching daemon 210 operates transparently from the perspective of the client server applications 120 that desire to access data from the directory server 110. To integrate the caching daemon 210 into the OS upon which these applications 120 are running, the SIA layer 220 is used to create a shared library with a predefined set of Application Programming Interfaces (APIs). The APIs correspond to the required set of security APIs that are used by the OS. By creating this shared library, it causes all of the existing libc security APIs to resolve to the new library. Accordingly, by using the SIA layer 220, all of the “security aware” applications 120 running on the OS will directly access the caching daemon 210 in a transparent manner, *i.e.*, the client

server applications 120 will not realize they are accessing the caching daemon 210, but rather will believe they are accessing the directory server 110 directly **[Dave, can you provide more details on this]**.

5 According to one embodiment of the present invention, the LDAP caching daemon 210 comprises a data cache 230, which is configured to store data that is retrieved from the directory server 110. When the application 120 binds to the caching daemon 210 and sends a data request over the connection 225, a controller 215 within the caching daemon 210 determines whether the requested data by the application 120 has been previously stored in the data cache 230. If the requested data has been previously cached, the controller 215 retrieves the requested data from the data cache 230 and sends this data to the requesting application 120 in the SIA layer 220 via the connection 225.

10 If, however, the requested data has not been previously stored in the data cache 230, then the controller 215 sends a request for the data to the directory server 110 via one of the plurality of connections established at 215. Once retrieved, the requested data is stored in the data cache 230 for any subsequent retrieval for a future request by another application 120 for the same data. The controller 215 of the caching daemon 210 then forwards the retrieved data from the directory server 110 to the requesting application 120 via the connection 225.

15 In accordance with one embodiment, the data cache 230 is indexed via two separate indexes, an ID hash index 310 and a name hash index 320 as shown in Figure 3. Accordingly, the information/data that is retrieved from the directory is stored once in the data cache 230, with multiple indexes into the stored data via the ID hash and name hash indexes 310, 320. According to one embodiment, the name hash index 320 is a hash table

20

that points to data entries in the data cache 230 whether these entries consist of group names or user names. Similarly, the ID hash index 310 is a hash table that points to entries in the data cache 230 whether the entries are user IDs or group IDs.

5 Turning now to Figure 4, a process 400 for facilitating access to the directory server 110 by a plurality of applications 120 running on the SIA layer 220 of the OS is provided. The process 400 commences at block 405 where the LDAP caching daemon 210 establishes a pool of connections 215 with the directory server 110. In accordance with one embodiment, the number of connections 215 established between the caching daemon 210 and the directory server 110 may be set by a parameter within the configuration file running on the OS. According to the illustrated embodiment, these connections 215 run continuously between the caching daemon 210 and the directory server 110, and awaits any data requests that are initiated by the application 120.

10 At block 410, the caching daemon 210 determines whether any of the applications 120 running on the OS are requesting information retrieval from the directory server 110. If any of these applications 120 are requesting information retrieval from the directory server 110, the caching daemon 210 determines at block 415 whether the information being requested by the application 120 has previously been downloaded into the data cache 230 of the caching daemon 210 from a previous request. If the information being requested by the application 120 has already been cached in the data cache 230 from the directory server 110, the caching daemon 210 accesses the previously cached information at block 420, and sends the information to the requesting application 120 at block 425 via the connection 225.

If the information being requested by the application 120 has not already been cached in the data cache 230 from a previous request, the caching daemon 210, via one of the plurality of connections 215, accesses the directory server 110 to retrieve the desired information at block 430. Upon downloading the requested information from the directory server 110, the caching daemon 210 stores the requested information in the data cache 230 at block 435 for any subsequent requests for the same information. At block 440, the caching daemon 210 sends the requested information to the application 120 that initially requested the data via the established connection 225.

The particular embodiments disclosed above are illustrative only, as the invention may be modified and practiced in different but equivalent manners apparent to those skilled in the art having the benefit of the teachings herein. Furthermore, no limitations are intended to the details of construction or design herein shown, other than as described in the claims below. It is therefore evident that the particular embodiments disclosed above may be altered or modified and all such variations are considered within the scope and spirit of the invention. Accordingly, the protection sought herein is as set forth in the claims below.

CLAIMS

WHAT IS CLAIMED:

1. A method for accessing a directory server, the method comprising:
establishing a first plurality of connections between the directory server and a caching daemon;
determining if an application is requesting information from the directory server;
determining if the requested information is stored in the caching daemon in response to determining that the application has requested information; and
sending the requested information to the application.
2. The method of claim 1, wherein determining if an application is requesting information from the directory server further comprises:
establishing a second connection between the application and the caching daemon;
and
receiving a request for information from the application over the second connection at the caching daemon.
3. The method of claim 1, further comprising:
retrieving the requested information from the caching daemon in response to determining that the caching daemon has the information stored therein.

4. The method of claim 1, further comprising:

retrieving the requested information from the directory server in response to determining that the caching daemon has not previously stored the information.

5. The method of claim 4, further comprising:

storing the information retrieved from the directory server at the caching daemon.

6. An apparatus, comprising:

a directory server for storing information; and

a caching daemon adapted to establish a first plurality of connections to the directory server, determine if an application is requesting information from the directory server, determine if the requested information is stored within the caching daemon; and send the requested information to the application.

7. The apparatus of claim 6, wherein the caching daemon is further adapted to establish a second connection with the application and receive a request for information from the application over the second connection.

8. The apparatus of claim 6, wherein the caching daemon comprises:

a data cache adapted to store a subset of the information stored in the directory server;
and

wherein the caching daemon is further adapted to retrieve the requested information from the data cache in response to determining that the requested information is part of the subset of information stored therein.

9. The apparatus of claim 6, wherein the caching daemon is further adapted to retrieve the requested information from the directory server in response to determining that the requested information is not stored within the data cache.

10. The apparatus of claim 9, wherein the caching daemon is further adapted to store the requested information retrieved from the directory server in the data cache.

11. An apparatus for accessing a directory server, the apparatus comprising:
means for establishing a first plurality of connections between the directory server and a caching daemon;
means for determining if an application is requesting information from the directory server;
means for determining if the requested information is stored in the caching daemon in response to determining that the application has requested information; and
means for sending the requested information to the application.

12. The method of claim 11, wherein the means for determining if an application is requesting information from the directory server further comprises:
means for establishing a second connection between the application and the caching daemon; and
means for receiving a request for information from the application over the second connection at the caching daemon.

13. The method of claim 11, further comprising:

means for retrieving the requested information from the caching daemon in response to determining that the caching daemon has the information stored therein.

14. The method of claim 11, further comprising:

means for retrieving the requested information from the directory server in response to determining that the caching daemon has not previously stored the information.

15. The method of claim 14, further comprising:

means for storing the information retrieved from the directory server at the caching daemon.

16. A caching daemon, comprising:

a data cache adapted to store a subset of information from a directory server; and
a controller adapted to establish a first plurality of connections to the directory server,
determine if an application is requesting information from the directory server,
determining if the requested information is stored in the data cache, and send
the requested information to the application.

17. The caching daemon of claim 16, wherein the controller is further adapted to establish a second connection to the application and receive a request for information from the application over the second connection.

18. The caching daemon of claim 16, wherein the controller is further adapted to retrieve the requested information from the data cache in response to determining that the data cache has the information stored therein.

19. The caching daemon of claim 16, wherein the controller is further adapted to retrieve the requested information from the directory server in response to determining that the data cache does not have the information stored therein.

20. The caching daemon of claim 19, wherein the controller is further adapted to store the information retrieved from the directory server at the data cache.

ABSTRACT OF THE DISCLOSURE

The present invention provides for a method and an apparatus for accessing a directory server. The directory server has information stored therein. A caching daemon establishes a first plurality of connections to the directory server. The caching daemon determines if an application is requesting information from the directory server over a second connection between the caching daemon and the application, and determines if the requested information is stored in a data cache within the caching daemon in response to determining that the application has requested information. If the requested information resides within the data cache, the caching daemon forwards the requested information to the application over the second connection. If the requested information is not present within the data cache, the caching daemon accesses the requested information from the directory server over one of the first plurality of connections. Upon receiving the requested information from the directory server, the caching daemon sends the requested information to the application over the second connection.

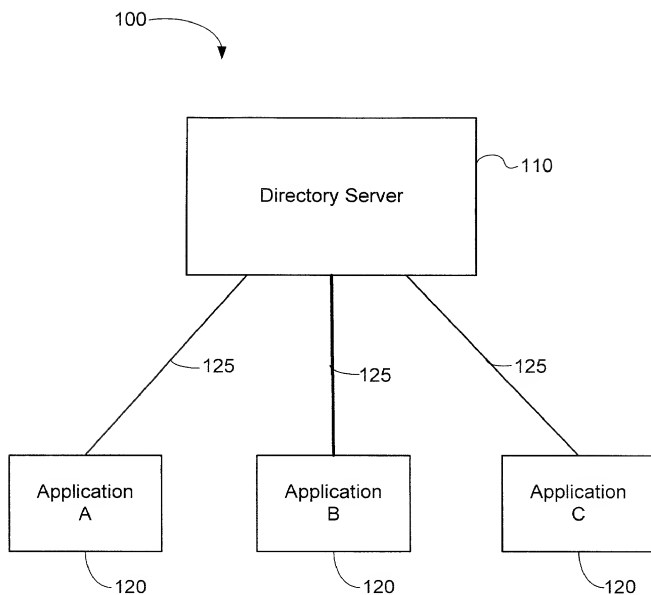
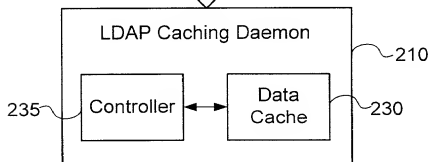
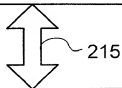
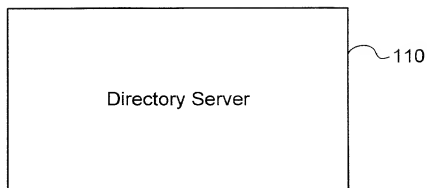


Figure 1
(Prior Art)

200

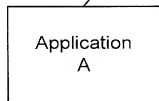


220

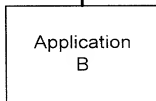
225

225

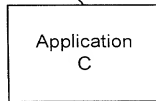
225



120



120



120

SIA Layer

Figure 2

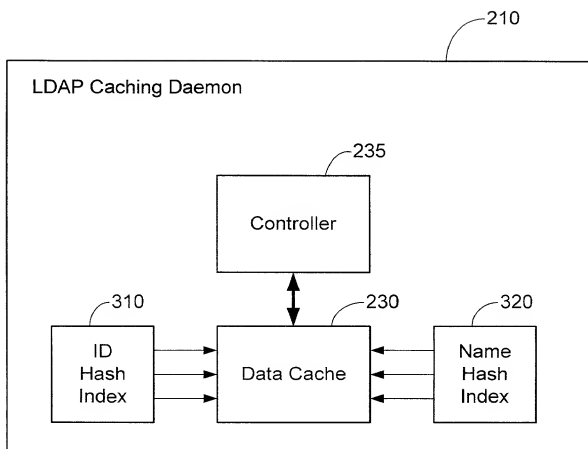


Figure 3

400

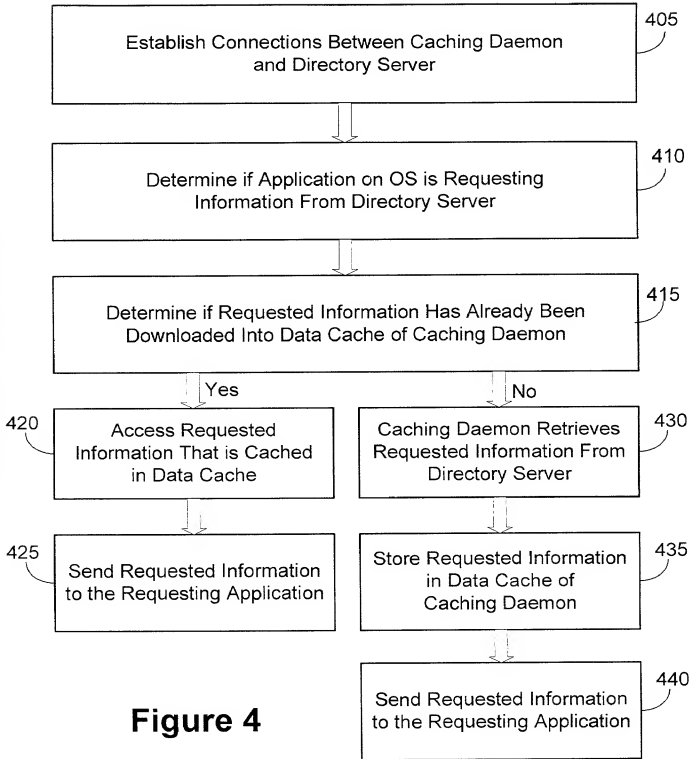


Figure 4

DECLARATION

SOLE/JOINT INVENTOR
ORIGINAL/SUBSTITUTE/CIP

As a below named inventor, I hereby declare that: my residence, post office address, and citizenship are as stated below next to my name. I believe I am the original, first, and sole inventor (if only one name is listed below) or a joint inventor (if plural inventors are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled **METHOD AND APPARATUS FOR PROVIDING ENHANCED ACCESS TO A LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL (LDAP) DIRECTORY SERVER**, as described in the specification attached.

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above, that I do not know and do not believe the same was ever known or used in the United States of America before my or our invention thereof, or patented or described in any printed publication in any country before my or our invention thereof or more than one year prior to this application; that the invention has not been patented or made the subject of an inventor's certificate issued before the date of this application in any country foreign to the United States of America on an application filed by me or my legal representative or assigns more than twelve months prior to this application; and that I acknowledge the duty to disclose information of which I am aware which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations § 1.56(a). Such information is material when it is not cumulative to information already of record or being made of record in the application, and

- (1) it establishes, by itself or in combination with other information, a prima facie case of unpatentability of a claim; or
(2) it refutes, or is inconsistent with, a position the applicant has taken or may take in:

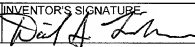
- (i) opposing an argument of unpatentability relied on by the Office, or
(ii) asserting an argument of patentability.

I hereby claim foreign priority benefits under Title 35, United States Code § 119 of any foreign application(s) for patent or inventor's certificates listed below and have also identified below any foreign application(s) having a filing date before that of the application(s) on which priority is claimed:

COUNTRY	APPLICATION NUMBER	DATE OF FILING	PRIORITY CLAIMED UNDER 35 USC 119
			<input type="checkbox"/> YES <input type="checkbox"/> NO

I hereby claim the benefit under Title 35 United States Code § 120 of any United States application(s) listed below and, insofar as any subject matter of any claim of this application is not disclosed in the prior United States Application, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations § 1.56(a) which occurred between the filing date of the prior application and the national PCT international filing date of this application.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

FULL NAME OF SOLE OR FIRST INVENTOR David J. Lindner		INVENTOR'S SIGNATURE 	DATE June 19, 2000
RESIDENCE 9 Stonybrook Rd., Nashua, NH 03063		CITIZENSHIP USA	
POST OFFICE ADDRESS			
FULL NAME OF SECOND JOINT INVENTOR		INVENTOR'S SIGNATURE	DATE
RESIDENCE		CITIZENSHIP	
POST OFFICE ADDRESS			
FULL NAME OF SOLE OR FIRST INVENTOR		INVENTOR'S SIGNATURE	DATE
RESIDENCE		CITIZENSHIP	
POST OFFICE ADDRESS			

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant/Patentee:
David L. Lindner

Filed: Concurrently herewith

Serial No.: Unknown

For: METHOD AND APPARATUS FOR
PROVIDING ENHANCED ACCESS TO A
LIGHTWEIGHT DIRECTORY ACCESS
PROTOCOL (LDAP) DIRECTORY SERVER

§ Group Art Unit:

§ Examiner:

§ Atty File: 2007.012900/P00-2935

§

POWER OF ATTORNEY BY ASSIGNEE

Under the provisions of 37 C.F.R. § 3.71, the undersigned assignee of record of the entire interest in the above-identified patent/patent application by virtue of an assignment recorded (check as applicable):

- ☒ Concurrently Herewith
☐ Date Recorded:
☐ Reel: Frame:

elects to conduct the prosecution of the application/maintenance of the patent to the exclusion of the inventor(s). The undersigned hereby declares that he has reviewed the above-referenced assignment and hereby declares that, to the best of his knowledge, title is in the Assignee, and further declares that all statements made herein of his own knowledge are true and that all statements made on information and belief are believed to be true. The assignee hereby revokes any previous powers of attorney and appoints the following to prosecute this application/maintain this patent and transact all business in the Patent and Trademark Office connected therewith:

Irene Kosturakis	33,724
Keith E. Lutsch	31,851
Joseph C. Arrambide	39,589
Sarah T. Harris	35,891
Louis Bruccleri	38,834
Richard P. Lange	27,296
Theodore S. Park	26,971

WILLIAMS, MORGAN & AMERSON

Danny L. Williams	31,892
Terry D. Morgan	31,181
J. Mike Amerson	35,426
Kenneth D. Goodman	30,460
Barbara S. Kitchell	33,928
Jeffrey A. Pyle	34,904
Randall C. Furlong	35,144
Scott F. Diring, Patent Agent	35,119
George J. Oehling, Patent Agent	40,471
Shelley P.M. Fussey, Patent Agent	39,456
Carolanne McAndrews, Patent Agent	44,914

Louis H. Iselin, Patent Agent 42,684
Raymund F. Eich, Patent Agent 42,508

Please direct all communications to: Williams, Morgan & Amerson, 7676 Hillmont, Suite 250,
Houston, Texas 77040 (713) 934-4050 attention of: Terry D. Morgan.

ASSIGNEE
COMPAQ COMPUTER CORPORATION

Date: 30 June 2000 BY: Irene Kosturakis
NAME: Irene Kosturakis
TITLE: Manager Worldwide Patent Development

Irene Kosturakis
Manager Worldwide Patent Development
Authorized To Sign This Document On Behalf Of
Compaq Computer Corporation
Pursuant To Board Of Directors Resolution
Date July 28, 1989